Bureau canadien
des brevets

Certification

Canadian Patent
Office

Certification

La présente atteste que les documents
ci-joints, dont la liste figure ci-dessous,
sont des copies authentiques des docu-
ments déposés au Bureau des brevets.

This is to certify that the documents
attached hereto and identified below are
true copies of the documents on file in
the Patent Office.

Specification and Drawings, as originally filed, with Application for Patent Serial No:
2,413,690, on December 6, 2002, by **IBM CANADA LIMITED-IBM CANADA
LIMITÉE**, assignee of Kyle Nathan Patrick, for "Zero Knowledge Document Comparison
between Mutually Distrustful Parties".

Agent certificateur/Certifying Officer

February 10, 2003

Date

Canada

ZERO KNOWLEDGE DOCUMENT COMPARISON
BETWEEN MUTUALLY DISTRUSTFUL PARTIES

5

ABSTRACT

Prior zero-knowledge protocols are used for exchanging secret keys, but not for comparing documents. The present invention provides a method of zero-knowledge document comparison between mutually distrustful parties by having each party exchange a set of random data and a shared hash function, applying the hash function to concatenations of the document and the sets of random data, and comparing the hashes.

10

CA9-2002-0058

ZERO KNOWLEDGE DOCUMENT COMPARISON
BETWEEN MUTUALLY DISTRUSTFUL PARTIES

Technical Field

5   The invention relates to the field of cryptography and more particularly to zero-knowledge methods for comparing documents between two parties.

Background Art

A zero-knowledge protocol, as in other types of interactive proofs, is a protocol between two

10   parties in which one party (the prover), tries to prove a fact to the other party (the verifier). The fact is typically secret information such as a password or, in cryptographic applications, the private key of a public key encryption algorithm. In zero-knowledge protocols, the prover can convince the verifier that he is in possession of the secret without revealing the secret itself. In particular, zero-knowledge protocols are cryptographic protocols in which: 1) the verifier cannot learn anything

15   from the protocol - no knowledge is transferred; 2) the prover cannot cheat the verifier and vice versa; and 3) the verifier cannot pretend to be the prover to any third party. Thus in a zero-knowledge protocol, the fact or secret itself, or any other useful information, is not revealed to the other party during the protocol, nor to any eavesdropper. The Fiat-Shamir protocol was the first practical zero-knowledge cryptographic protocol.

20   Hash functions are commonly used in cryptography. A one-way hash function is a function that takes a variable-length input string and converts it into a fixed-length output string. An example of such a hash function is the SHA-1 function. It is impossible to determine the input string from the hashed string.

In some situations where A and B are two distrustful parties, it may be necessary for the

25   parties to learn whether two documents which are possessed by the respective parties are the same or substantially the same. For example, B may claim to have a copy of A's secret document and A's course of action may hinge on whether B's claim is true. Neither party however can disclose their respective document to the other in order to verify B's claim without destroying their secrecy. While the use of zero-knowledge protocols is known for exchanging secret keys it has not been used for

30   comparing documents.

There is a need therefore a strong zero-knowledge document comparison method between mutually distrustful parties.

Disclosure of Invention

The present invention therefore provides a method of securely comparing a first document in possession of a first party and a second document in possession of a second party, without revealing the contents of the first document to the second party or the contents of the second

5   document to the first party, said method comprising the steps of:

i) said first and second parties each generating its own set of random data;

ii) each party exchanging the set of random data and a shared hash function with the other party;

iii) each party computing a first value consisting of the output of the shared hash function

10   where the input to the hash function is the consecutive concatenation of the document in each party's possession, followed by that party's set of random data, followed by the other party's set of random data;

iv) each party computing a second value consisting of the output of the shared hash function where the input to the hash function is the consecutive concatenation of the document in each

15   party's possession, followed by the other party's set of random data, followed by that party's set of random data;

v) each party sending its first value to the other party and receiving the other party's first value; and

vi) each party comparing the other party's first value to its second value;

20   vii) each party concluding that if the values are the same, then the two documents are the same, but that otherwise the two documents are different.

The invention further provides a computer program product and an article for carrying out the method.

25

Brief Description of Drawings

In drawings which disclose a preferred embodiment of the invention:

Fig. 1 is a schematic illustration of a computer network according to the present invention;

30   and

Fig. 2 is a flow chart illustrating the method of the invention.

Best Mode(s) For Carrying Out the Invention

With reference to Fig. 1, a communications link, such as a computer network, is designated generally as 10. Parties A and B, who distrust each other, communicate between their respective computers 12 and 14, which have central processors and are capable of generating random numbers, and comparing numbers. A possesses a document containing information, in electronic form or otherwise, referred to as document1. B possesses a document containing information, in electronic form or otherwise, referred to as document2. Parties A and/or B, would like to take some further action only if one or the other or both can be assured that they both have the same document. They may not care to know each other's identity.

If the respective documents, document1 and document2, are not already in the form of a bit string, they are scanned or otherwise converted to that format. Next, A sends B a collection of random bits, Ra, preferably incorporating a timestamp. B sends A a collection of random bits, Rb, preferably incorporating a timestamp. A compares Ra to Rb and aborts the comparison if they are the same, since the comparison will only work if the random numbers generated by A and B are different. Similarly B compares Rb to Ra and aborts the comparison if they are the same. They will then restart and generate fresh random numbers if they wish to continue.

Once A and B have exchanged non-identical random strings Ra and Rb, and have agreed on one-way hash functions H_1, H_2, A computes firstValueA by concatenating document1 with Ra and Rb, in that order, to form a string document1 + Ra + Rb, in that order and then applying to that string a one-way hash function H_1. Any suitable cryptographic one-way hash function, such as the SHA-1 function, may be used. A then computes secondValueA by concatenating document1 + Rb + Ra, in that order, and applying to it one-way hash function H_2. Similarly B computes firstValueB by concatenating document2 with Rb and Ra, in that order, to form a string document2 + Rb + Ra, in that order, and then applying to that string one-way hash function H_2. B then computes secondValueB by concatenating document2 with Ra and Rb, in that order, to form a string document2 + Ra + Rb, in that order, and then applying to that string a one-way hash function H_1. Hash functions H_1 and H_2 may be the same.

It has been agreed upon beforehand that A will transmit the encrypted string firstValueA first to B, although the method will work regardless of which party sends the encrypted string to the other first. Upon completion of the foregoing steps, A sends B a message indicating that it has computed firstValueA and secondValueA, and either before, after, or at the same time as A sends that message, B sends A a message indicating that it has computed firstValueB and secondValueB. A then sends B firstValueA. B sends A firstValueB immediately upon receipt of A's firstValueA. If A does not

CA9-2002-0058   3

receive B's firstValueB within a few milliseconds (in the absence of some other explanation such as a communication breakdown), A knows B did not have the same document and is trying to gain an advantage over A.

If A receives B's firstValueB in a timely way, A compares the received firstValueB with its own secondValueA. B also compares the received firstValueA with its own secondValueB. If the comparisons fail, then A and B know they statistically have different documents, and if the comparison does not fail, then statistically they have the same document. With that knowledge they may then proceed with their intended actions, or not.

Such comparisons may allow for a certain statistical dissimilarity in the strings or range of equivalence. A strict application of a hash function such as SHA-1 to a bit stream, such as a document, will produce a value that is statistically impossible to produce by supplying a second different meaningful bit stream. A strict application of the hash function does not allow for variance resulting from transmission errors or conversion between formats. Such variances would typically result in different hash codes. However, it is possible to describe a process where minor variation in the source can be handled. A document may be normalized before being passed to a hash function, or a hash function could be constructed that handles the normalization internally as part of the implementation. In this way inconsequential differences in the documents such as case type and spacing can be ignored.

For example, the parties could agree that whitespace (such as spaces, tabs and carriage returns) and character case are insignificant. The document could then be converted to a normalized form where there is no whitespace and all the characters are lowercase. The other approach would be to make the hash function ignore whitespace and change characters to lowercase before injection into the rest of the algorithm.

Thus it will be seen that according to this method, A and/or B cannot prove anything to a third party without revealing documents. A and B do not exchange the actual documents or hashed documents. Further, A or B cannot fool another party C into thinking it has the document by mirroring, resending or replaying the hash received from the other party to the third party. B cannot assert computational delay as they have previously asserted a pre-computation.

The present invention is described above as a computer-implemented method. It may also be embodied as a computer hardware apparatus, computer software code or a combination of same. The invention may also be embodied as a computer-readable storage medium embodying code for implementing the invention. Such storage medium may be magnetic or optical, hard or floppy disk, CD-ROM, firmware or other storage media. The invention may also be embodied on a computer

readable modulated carrier signal.

As will be apparent to those skilled in the art in the light of the foregoing disclosure, many alterations and modifications are possible in the practice of this invention without departing from the spirit or scope thereof. Accordingly, the scope of the invention is to be construed in accordance with the substance defined by the following claims.

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A method of securely comparing a first document in possession of a first party and a second
5   document in possession of a second party, without revealing the contents of the first document to
the second party or the contents of the second document to the first party, said method comprising
the steps of:

i) said first and second parties each generating its own set of random data;

ii) each party exchanging said set of random data and a shared hash function with the other
10   party;

iii) each party computing a first value consisting of the output of said shared hash function
where the input to the hash function is the consecutive concatenation of the document in each
said party's possession, followed by that party's set of random data, followed by the other
party's set of random data;

15   iv) each party computing a second value consisting of the output of said shared hash function
where the input to the hash function is the consecutive concatenation of the document in each
said party's possession, followed by the other party's set of random data, followed by that
party's set of random data;

v) each party sending its first value to the other party and receiving the other party's first
20   value; and

vi) each party comparing said other party's first value to its second value;

vii) each party concluding that if the said values are the same, then the two documents are
the same, but that otherwise said two documents are different.

25   2.. The method according to claim 1 further comprising the steps of:

viii) after computing said first and second values according to steps iii) and iv) above, each
said first and second parties sending confirmation to the other party that each said party's
first and second values have been computed, and waiting for said confirmation from said
other party that each said party's first and second values have been computed before
30   proceeding; and

ix) after one party has sent its first value to the other party according to step v) above,
aborting the comparison if the other party does not respond with its first value within a pre-
determined length of time.

3.. The method according to claim 2 further comprising the steps of:

x) after step i) and before step ii), each party examining the other party's set of random data for suitability and aborting the comparison if suitability is not established.

5   4. The method according to claim 3 wherein said other party's random data is determined to be unsuitable if it is identical to said examining party's set of random data.

5. The method according to claim 1 wherein said parties exchange two shared hash functions, a first hash function applied by said first party in step iii) and said second party in step iv) and a second

10   hash function applied by said second party in step iii) and said first party in step iv).

6. The method according to claim 1 wherein said documents are normalized prior to computation of said first and second values to allow the method to ignore inconsequential differences between said documents.

15

7. The method according to claim 1 wherein said hash function is adapted to act on said documents in a normalized way to allow the method to ignore inconsequential differences between said documents.

20   8. A computer program product for securely comparing a first document in possession of a first party and a second document in possession of a second party, without revealing the contents of the first document to the second party said computer program product comprising:

a computer usable medium having computer readable program code means embodied in said medium for:

25   i) generating a set of random data for said first party;

ii) exchanging said set of random data and a shared hash function with the other party;

iii) computing a first value consisting of the output of said shared hash function where the input to the hash function is the consecutive concatenation of the document in each said party's possession, followed by that party's set of random data, followed by the other party's

30   set of random data;

iv) computing a second value consisting of the output of said shared hash function where the input to the hash function is the consecutive concatenation of the document in each said party's possession, followed by the other party's set of random data, followed by that party's

set of random data;

v) sending said first value to the other party and receiving the other party's first value; and

vi) comparing said other party's first value consisting of the output of said shared hash function where the input to the hash function is the consecutive concatenation of the document in said other party's possession, followed by said set of random data, followed by the other party's set of random data, to its second value;

vii) concluding that if the said values are the same, then the two documents are the same, but that otherwise said two documents are different.

9. The computer program product of claim 8 wherein said computer usable medium further has computer readable program code means embodied in said medium for:

viii) after computing said first and second values according to iii) and iv) above, sending confirmation to the other party that the first and second values have been computed, and waiting for confirmation from said other party that said other party's first and second values have been computed before proceeding; and

ix) after sending its first value to the other party according to v) above, aborting the comparison if the other party does not respond with its first value within a pre-determined length of time

10. The computer program product of claim 9 wherein said computer usable medium further has computer readable program code means embodied in said medium for:

x) after step i) and before step ii) examining the other party's set of random data for suitability and aborting the comparison if suitability is not established.

11. The computer program product of claim 10 wherein said other party's random data is determined to be unsuitable if it is identical to said examining party's set of random data.

12. The computer program product of claim 8 wherein said parties exchange two shared hash functions, a first hash function applied by said first party in step iii) and said second party in step iv) and a second hash function applied by said second party in step iii) and said first party in step iv)

13. The computer program product of claim 8 wherein said documents are normalized prior to computation of said first and second values to allow the method to ignore inconsequential

differences between said documents.

14. The computer program product of claim 8 wherein said hash function is adapted to act on said documents in a normalized way to allow the method to ignore inconsequential differences between
5   said documents.

15. An article comprising:
        a computer readable modulated carrier signal;
        means embedded in said signal for securely comparing a first document in possession of a
10   first party and a second document in possession of a second party, without revealing the
        contents of the first document to the second party by:
        i) generating a set of random data for said first party;
        ii) exchanging said set of random data and a shared hash function with the other party;
        iii) computing a first value consisting of the output of said shared hash function where the
15   input to the hash function is the consecutive concatenation of the document in each said
        party's possession, followed by that party's set of random data, followed by the other party's
        set of random data;
        iv) computing a second value consisting of the output of said shared hash function where the
        input to the hash function is the consecutive concatenation of the document in each said
20   party's possession, followed by the other party's set of random data, followed by that party's
        set of random data;
        v) sending said first value to the other party and receiving the other party's first value; and
        vi) comparing said other party's first value consisting of the output of said shared hash
        function where the input to the hash function is the consecutive concatenation of the
25   document in said other party's possession, followed by said set of random data, followed by
        the other party's set of random data, to its second value;
        vii) concluding that if the said values are the same, then the two documents are the same,
        but that otherwise said two documents are different.

30   16. The article of claim 15 wherein said signal further has means embodied therein for:
        viii) after computing said first and second values according to iii) and iv) above, sending
        confirmation to the other party that each said party's first and second values have been
        computed, and waiting for said confirmation from said other party that said party's first and

second values have been computed before proceeding; and

ix) after sending its first value to the other party according to v) above, aborting the comparison if the other party does not respond with its first value within a pre-determined length of time.

5

17. The article of claim 16 wherein said signal further has means embodied therein for

x) after step i) and before step ii) examining the other party's set of random data for suitability and aborting the comparison if suitability is not established.

10   18. The article of claim 17 wherein said other party's random data is determined to be unsuitable if it is identical to said examining party's set of random data.

19. The article of claim 15 wherein said parties exchange two shared hash functions, a first hash function applied by said first party in step iii) and said second party in step iv) and a second hash

15   function applied by said second party in step iii) and said first party in step iv).

20. The article of claim 15 wherein said documents are normalized prior to computation of said first and second values to allow the method to ignore inconsequential differences between said documents.

20

21. The article of claim 15 wherein said hash function is adapted to act on said documents in a normalized way to allow the method to ignore inconsequential differences between said documents.
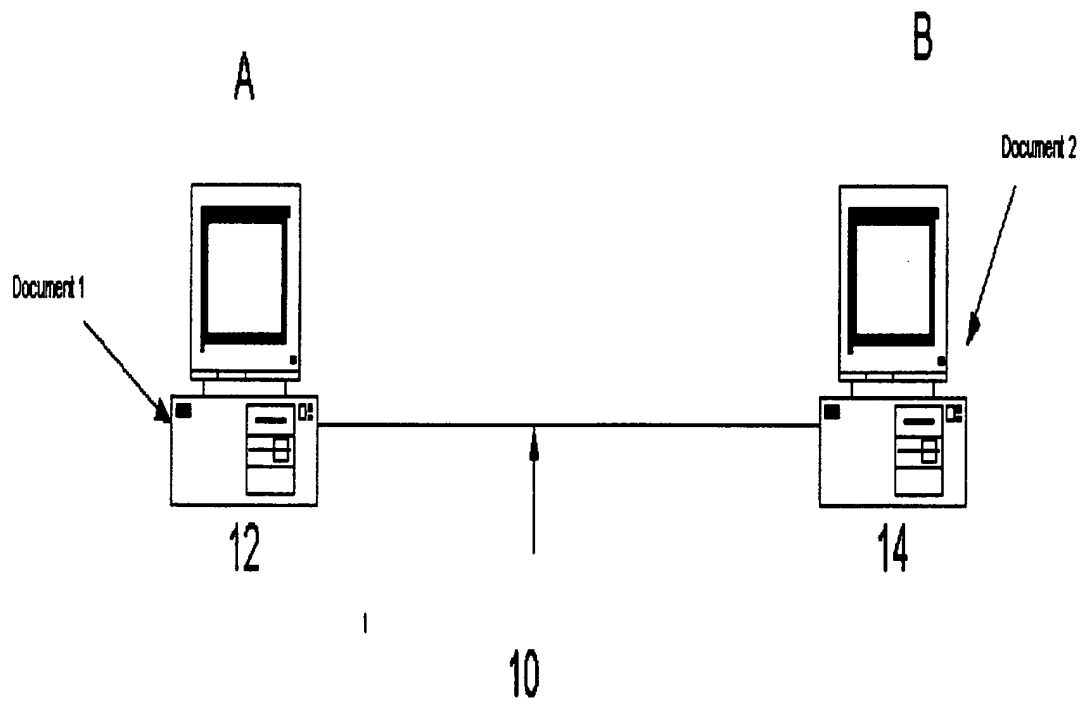
A

B

Document 2

Document 1

12

14

10

FIG. 1

FIG. 2

A sends B random string, Ra and B sends A random string, Rb

A compares Ra to Rb and B compares Rb to Ra

Are Ra and Rb the same?

Yes

No

A computes hash(document1 + Ra + Rb) = firstValueA

A computes hash(document1 + Rb + Ra) = secondValueA

B computes hash(document2 + Rb + Ra) = firstValueB

B computes hash(document2 + Ra + Rb) = secondValueB

A sends B a message indicating that it has computed firstValueA and secondValueA and B sends A a message indicating that it has computed firstValueB and secondValueB

A sends B firstValueA

Has B sent first ValueB within time t of A's transmission?

NO

STOP

YES

A compares the received firstValueB with its own secondValueA and B compares the received firstValueA with its own secondValueB

[First ValueB = secondValueA] and [FirstValueA = secondValueB] ?

NO

STOP

YES

Proceed